Universitatea POLITEHINICA Bucureşti
Facultatea de Automatică şi Calculatoare

Managementul Proiectelor Software

Hierarchy Analysis
Software Development Document
v 1.0

Echipa Farik

Bucureşti, 2012

0. Cuprins

1. Scopul documentului

Acest document are rolul de a descrie soluţia tehnică proiectată pentru componenta *Hierachy Analysis* din cadrul sistemului OCR modular pe care studenţii îl dezvoltă în cadrul cursului de Managementul Proiectelor Software din cadrul Facultăţii de Automatică şi Calculatoare.

Documentul serveşte drept ghid pentru implementarea soluţiei software pentru modulul *Hierarchy Analysis*.

2. Vedere de ansamblu a soluției proiectate

2.1. Fond

Functiile si scopul acestui proiect sunt redactate in Documentul Specificatiilor de Proiect la adresa http://elf.cs.pub.ro/mps/wiki/proiect/descriere.

2.2. Context si design de bază

Flow-ul principal este urmatorul:utilizatorul încarcă unul sau mai multe fișiere XML obținute în urma rulării analizei de layout și ruleaza un analizor ierarhic pentru acestea sau să încarca un fișier XML care conține rezultatul analizei ierarhice.

2.3. Decizii de implementare

Componenta dezvolatata va fi implementata in limbajul Java.

Versiunea de Java folosita va fi Java 7.

IDE-urile folosite pentru dezvoltarea aplicatiei sunt Eclipse (Indigo) si NetBeans (7).

Aceasta decizie a fost luata pentru ca Java este gratuit, portabil, rapid si sigur.

2.4. Formatul fișierelor utilizate

Aplicatie foloseste fisiere XML ca modalitate de introducere a datelor in sistem. Datele de iesire sunt tot fisiere XML.

Toate fisierele XML cu care lucreaza componenta *Hierarchy Analysis* sunt validate de catre XSD-urile pus la dispozitie la aceasta adresa http://elf.cs.pub.ro/mps/wiki/proiect/specificatii-xsd.

3. Modelul datelor

3.1. Clasa HierarchyAnalysis

Scopul acestei clase este obtinerea unui fisier XML care contine rezultatul analizei ierarhice. Ii permite utilizatorului sa si aleaga modul in care va obtine fisierul de output XML.

Prima varianta: incarca unul sau mai multe fisiere XML obtinute in urma analizei de layout si ruleaza un analizor ierarhic pentru acestea. In urma rularii va obtine fisierul cu rezultatul analizei ierarhice.

Cea de a doua posibilitate este incarcarea directa a fisierului de catre utilizator si este cea recomandata.

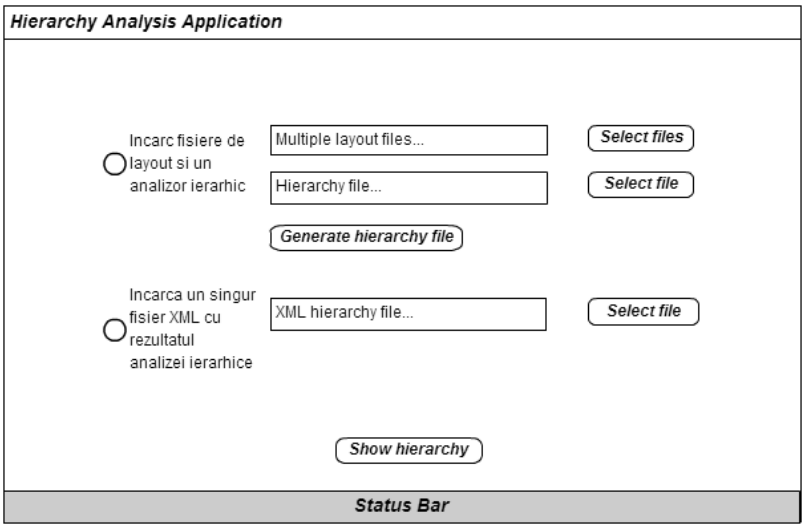
3.1.1. Structura clasei

Membrii clasei sunt urmatorii:

- Un grup de elemente RadioButton care permite utilizatorului selectia unui anumit mod de introducere a datelor. Exista 2 moduri posibile pentru a obtine fisierul de intrare :
  1. Selectia mai multor fisiere XML de layout si a unui analizor ierarhic. Pentru a selecta fisierele XML se va folosi JfileChooser-ul : BrowseXMLs, iar pentru selectia analizorului ierarhic se va folosi JFileChooser-ul : BrowseExec.
  2. Selectia unui singur fisier XML care contine rezultatul analizei ierarhice cu ajutorul unui JFileChooser numit : BrowseUniqueXML.

- Un JButton: ShowHierarchy. La apasarea acestui buton se va inchide fereasta curenta si se va deschide o noua fereastra care va afisa ierarhia si va permite utilizatorului sa corecteze eventualele erori.

3.1.2. Diagrama UI a clasei



Validarea fisierelor XML de intrare se face dupa incarcarea acestora. Operatia este realizata in background, utilizatorul fiind instiintat daca operatia a avut succes cu ajutorul unui StatusBar, in care mesajele vor fi : Validare in curs, Fisiere XML valide sau Fisiere XML invalide.

3.2. Clasa Editor

Scopul acestei clase este afisarea diferitelor tipuri de blocuri, unirea lor , spargerea lor sau recatalogarea diferitelor blocuri.

3.2.1. Structura clasei

Membrii clasei sunt urmatorii :

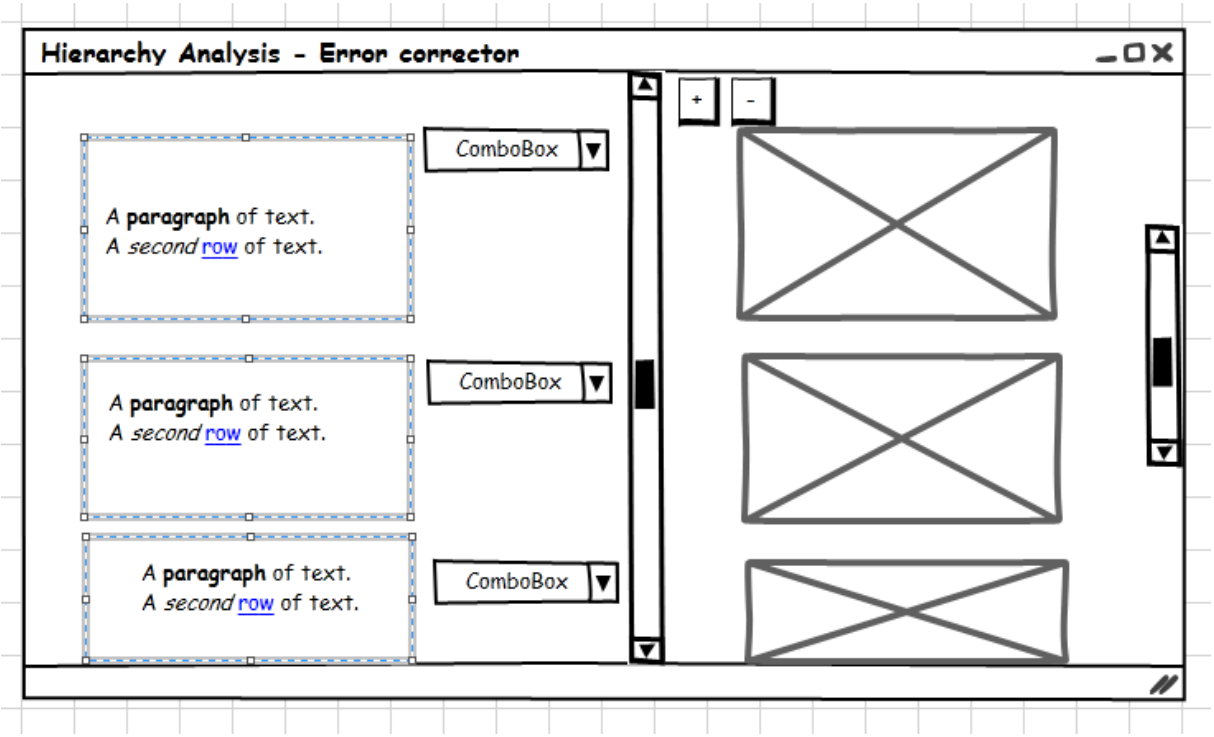
Clasa va contine doua Panel-uri. Cel din stanga va contine o suprafata reprezentand pagina, pe care se vor afisa blocurile de text si tipul acestora. In dreapta fiecarui bloc se vor afla 2 ComboBox-uri, unul pentru a permite editarea respectivului bloc : split, merge, schimbare tip si celalalt pentru schimbarea fontului blocului.

Panel-ul din dreapta va contine o suprafata reprezentand pagina, pe care se vor afisa imaginile ce au fost prelucrate de catre analizatorul de layout. Daca sunt inspectate mai multe imagini , acestea vor putea fi afisate una deasupra celelalte in panel ul acesta. Imaginile sunt afisate in cadrul unui JScrollPane. De asemenea, exista 2 butoane care permit zoom-ul pe o anumita pagina. Pentru cazul in care zoom-ul este prea mare, ea va putea fi inspectata cu ajutorul unui JScrollBar orizontal.

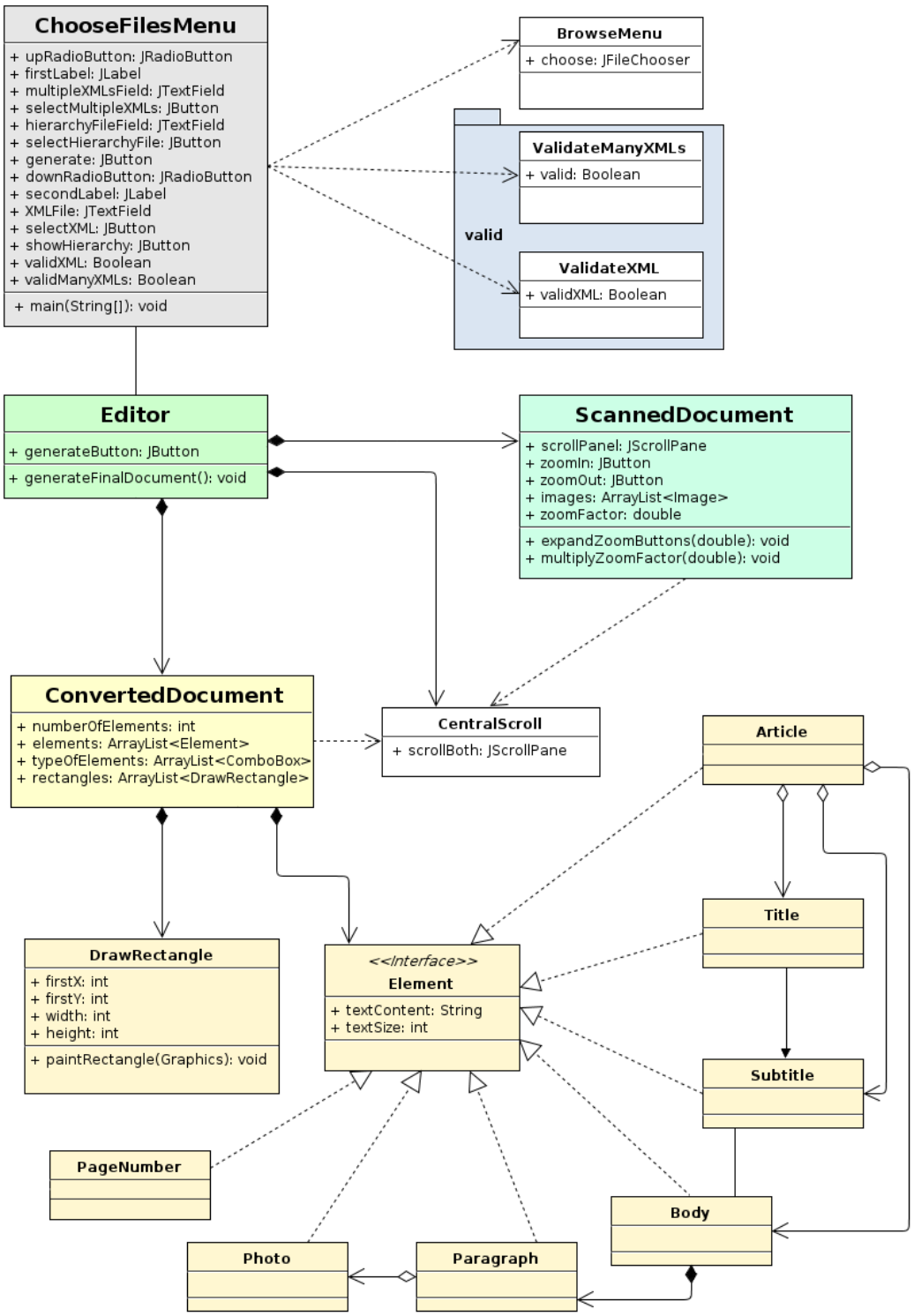
In centrul ecranului se va afla un JscrollBar care va derula in mod sincron cele doua emisfere. In cazul unei desincronizari pe verticala, exista in partea dreapta un alt JscrollBar prin care se va face potrivirea.

Blocurile din partea stanga sunt instante ale unei clase parinte Element. Blocurile pot fi de tipul: Paragraph, Title, Subtitle, Article, Body, PageNumber, Image. Caracteristicile comune tuturor blocurilor sunt: existenta unei instante a clasei Cadru (vizibila in interfata grafica printr un poligon incadrator) si continutul propriu-zis de informatie.

3.2.2. Diagrama UI a clasei



3.2.3. Diagrama UML a clasei



4. Structuri de date folosite

Anexa privind structurile de date folosite se gaseste la aceasta adresa https://docs.google.com/file/d/0B8wpiA-CzQ4-MjlvdUR4Qm5hNFU/edit.

5. Testare

5.1. Planul de testare:

https://docs.google.com/file/d/0B8wpiA-CzQ4-Z25zQlJvdS1FdVE/edit

5.2. Managementul bug-urilor:

https://docs.google.com/spreadsheet/ccc?key=0AgrO3KayjJXQdDMwbmVVUTg2ZlFsb1hyVk5yZUZkWEE#gid=0

5.3. Workflow de fixarea bug-urilor:

- Un fisier partajat in care se va raporta fiecare bug gasit de catre QE-s
- Fiecare DEV va prelua bug-urile asignate lui si le va fixa (in ordinea severitatii)
- Dupa fixare QE-ul va face regresie la bug

6. Cerințe de sistem

Procesor: Intel Pentium IV, 2 GHz

RAM: 2GB

Sistem de operare: Windows XP, Windows Vista, Windows 7, Windows 8, MAC OS X, Linux 2.4

Runtime environments: JRE 1.7

Conexiune la Internet: nu este necesara.